Remarks

Applicants respectfully request reconsideration of the present application in view of the foregoing amendments and the following remarks. Claims 1-30 are pending in the application. Claims 1-30 are rejected. No claims have been allowed. Claims 1, 13, 18, 21, and 27 are independent.

Cited Art

The Action cites Pinter et al. (6,457,023) ("Pinter").

Telephonic Interview

Applicant wishes to thank Examiner Chavis for extending a telephonic Examiner Interview on October 26, 2007. Claims 22, 23, 26 and Pinter were discussed. Although specific agreement was not reached, Applicant now presents reasoned arguments in light of the Examiner's description of Pinter and the rejection of claims 23 and 26 in the Office Action of July 25, 2007 (hereinafter Action).

In the interest of reaching a shared understanding of the disclosure of Pinter, Applicant makes the following observations.

Pinter describes "static evaluation of the lifetime of objects allocated in memory…which find application in reducing the overhead of generational garbage collection." [Pinter, abstract.] All of the objects in the program are detected, and then lower, upper and "middle" estimations of their object lifetimes are given to determine which objects should be "promoted to an older generation," i.e., collected less often. [*Id.,* 1:13-27; 4:9--65.]

Such object analysis is performed for the entire program. First, all the objects are detected. This process involves building a control flow graph of the entire program and then performing a reachability analysis **for each** instruction in the program. "[A] control flow graph is constructed for each analyzable element." [*Id.*, 8:34-38; Fig. 4 at 42.] Then, "a reachability analysis is performed *for each* instruction in the program, as well as at the entrance to and exit from each basic block," producing a reachability graph. [*Id.* at 4:43-50; 8:42-44; Fig.4 at 46.] This reachability graph represents a specific sort of reachability-- "the references between live variables and objects generated at the allocation sites." [*Id.* at 8:42-44.]

Next, "*for each* allocation site representing the generation of a particular object, the lifetime of the object is statically estimated…." [*Id.*, 8:50-52; Fig.4 at 48.]

Further, this analysis only happens going forward. That is, Pinter is interested in object lifetimes, starting with the birth of the object, and continuing until garbage collection.

Claim Rejections under 35 USC § 102

The Action rejects claims 1-21 and 27-29 under 35 USC 102(e) as being anticipated by Pinter. Applicants respectfully submit the claims are allowable over the cited art. For a 102(e) rejection to be proper, the cited art must show each and every element as set forth in a claim. (*See* MPEP § 2131.01.) However, the cited art does not describe each and every element. Accordingly, applicants request that all rejections be withdrawn. Claims 1, 13, 18, 21, and 27 are independent.

Claim 1.

Amended claim 1 reads as follows:

> A static data flow analysis method comprising:
> receiving a start state from a user;
> chasing a data flow instance of the start state through a first data flow graph local to a first procedure until a procedure transition instruction is encountered;
> resolving the transition instruction to a second procedure pointed to by a call graph; and
> chasing the data flow instance into a data flow graph local to the second procedure.

Pinter, as applicant understands, does not teach or suggest the amended claim 1 language "receiving a start state from a user." For at least this reason, a 102(e) rejection is improper, and claim 1 is currently in condition for allowance.

Further Pinter teaches against such language. As mentioned in the Pinter discussion section, above, the entire program is analyzed. The summary of the invention in Pinter states "Preferred embodiments of the present invention teach an improved technique of data flow analysis using pointer analysis, which can be applied to improve the performance of generational garbage collection of heap memory." [Pinter, 3:36-39.] Analyzing an entire program, and using the results of the analysis for garbage collection teaches away from the claim 1 language "receiving a start state from a user;" as there is no need for any further input, such as a start state from a user, to analyze an entire program for heap memory garbage collection.

Pinter also fails to teach or suggest, and, indeed, teaches away from the amended claim 1 language "chasing a data flow instance of the start state *through a first data flow graph local to a first procedure*." As Pinter does not teach or suggest a user-defined start state, it also does not teach or suggest the additional language "chasing a data flow instance of the start state." Pinter does not teach chasing individual data flow instances (such as a start state data flow instance) as far as Applicant can tell. Rather, with its insistence on performing analysis on the entire program ("At each point in the code, a reachability graph is constructed," Pinter, 4:41-42, "a reachability analysis is performed for each instruction in the program," *Id. at*, 8:43-44) Pinter further leads away from chasing a data flow instance of the start state.

Further, Pinter does not teach or suggest the italicized claim language "chasing a data flow instance of the start state through a data flow graph *through a first data flow graph local to a first procedure* …" Nowhere does Pinter discuss data flow graphs local to procedures. In fact, the Pinter analysis discusses procedures only in reference to program call graphs, as reproduced below:

> Interprocedural data flow analyses make use of the program call graph (PCG), which is a flow multigraph in which each procedure is represented by a single node and in which an edge represents a potential call of a procedure from a call site of another procedure. In the presence of function pointers or virtual methods a call site may contribute multiple edges to different procedures. Pinter, 3:20-27.

This does not teach or suggest a data flow graphs local to a procedure. Rather, it teaches a procedure being "represented by a single node." A single node representation of a procedure teaches away from "a flow graph local to a procedure." For this further reason, claim 1 is in condition for allowance.

Moreover, Pinter does not teach or suggest "chasing the data flow instance into a data flow graph local to the second procedure." The Action in its rejection of the original language cites to the following passage in Pinter:

> According to another aspect of the invention, the step of determining the second point is performed by pointer analysis. The pointer analysis includes the steps of constructing a control flow graph for an analyzable element of the program. constructing a hierarchy graph for each analyzable element, and constructing a call graph for the analyzable elements. Responsive to the control flow graph, the hierarchy graph and the call graph, a reachability graph is constructed for the analyzable elements. Pinter 5:24-32.

This passage in Pinter does not mention a graph local to a procedure, and, further, does not discuss chasing a data flow instance into such a graph.

For all these reasons Applicant submits that claim 1 is in condition for allowance.

Claims 2-12.

Claims 2-12 ultimately depend on claim 1. Thus, at least for the reasons set forth above with respect to claim 1, claims 2-12 should also be in condition for allowance. These claims also set forth independently patentable combinations of method acts.

Claim 13.

Claim 13 is rejected by the action as follows: "In reference to claims 13-14 and 18-19, see the rejection of claims 1-2. [Action, page 4.] To not belabor the point, using the same analysis as given for claim 1 claim 13 is patentable.

Further, Pinter fails to teach or suggest, e.g., the following claim 13 features:

- receiving binary code and *a start state*;
- creating a data flow graph for a procedure comprising *the start state*, the data flow graph comprising pointers to instructions in the procedures and instructions representation;
- chasing *a data instance of the start state* through instructions in the data flow graph *corresponding to states in a state machine*;

Initially, it is noted that the Office has not identified where in Pinter an alleged anticipatory teaching of "receiving … *a start state*" is to be found. The Office also has not identified where the in Pinter "creating a data flow graph for a procedure comprising *the start state* …" and "chasing *a data instance of the start state*" are located, either.

Claim 13 is rejected by the action as follows: "In reference to claims 13-14 and 18-19, see the rejection of claims 1-2." Action, page 4. However, the rejections for claims 1 and 2 also do not include alleged anticipatory teachings of the above-quoted claim language. *See* Action, pages 2 and 3.

As such, the Office has failed to make a prima facie case of anticipation against claim 13, and claim 13 is therefore allowable. Further, after carefully reading Pinter, Applicant cannot locate in Pinter, at a minimum, "a start state," let alone "receiving … *a start state,*". "creating a data flow graph for a procedure comprising *the start state, …*" and "chasing *a data instance of the start state.*"

Moreover, Pinter teaches against such language. The summary of the invention in Pinter states "Preferred embodiments of the present invention teach an improved technique of data flow analysis using pointer analysis, which can be applied to improve the performance of generational

garbage collection of heap memory." [Pinter, 3:36-39.] Analyzing an entire program, and using the results of the analysis for garbage collection teaches away from the claim 1 language "receiving a start state;" as there is no need for any further input, such as a start state, to analyze an entire program for heap memory garbage collection. For at least this reason, claim 13 is in condition for allowance.

In the event that the Office maintains the rejection of independent claim 13 under 35 U.S.C. §102, Applicant respectfully requests that the Office, in the interests of compact prosecution, identify on the record and with specificity sufficient to support a prima facie case of anticipation, where in the Pinter patent the subject feature of independent claim 13 of "receiving … *a start state,*". "creating a data flow graph for a procedure comprising *the start state,* …" and "chasing *a data instance of the start state."* are alleged to be taught.

Claim 14.

Pinter fails to teach or suggest the following claim 14 features:

- a pointer dereference in an instruction in a data flow graph corresponding to a state in the state machine representing a pointer dereference table,
- performing a backward recursive search indicated in the pointer dereference table according to the addressing mode of the pointer dereference,
- and identifying a location in the backward recursive search.

Initially, it is noted that the Office has not identified where in Pinter alleged anticipatory teachings of the above claim features are to be found. Claim 14 is rejected with reference to claims 1 and 2. However, claims 1 and 2 do not mention the above-bulleted claim features. As such, the Office has failed to make a prima facie case of anticipation against claim 14, and claim 14 is therefore allowable. Further, after carefully reading Pinter, Applicant cannot locate in Pinter, at a minimum, the claim language quoted above.

In the event that the Office maintains the rejection of independent claim 14 under 35 U.S.C. §102, Applicant respectfully requests that the Office, in the interests of compact prosecution, identify on the record and with specificity sufficient to support a prima facie case of anticipation, where in the Pinter patent the subject features of independent claim 14 of "upon encountering a pointer dereference in an instruction in a data flow graph corresponding to a state in the state machine representing a pointer dereference table," "performing a backward recursive search indicated in the pointer dereference table according to the addressing mode of the pointer dereference," and "identifying a location in the backward recursive search" are alleged to be

taught. Moreover, claim 4 ultimately depends on claim 13. Thus, at least for the reasons set forth above with respect to claim 13, claim 4 should also be in condition for allowance.

Claims 15-17.

Claims 15-17 ultimately depend on claim 13. Thus, at least for the reasons set forth above with respect to claim 13, claims 15-17 should also be in condition for allowance. These claims also set forth independently patentable combinations of method acts.

Claim 18.

Claim 18 is rejected by the action as follows: "In reference to claims 13-14 and 18-19, see the rejection of claims 1-2." [Action, page 4.] To not belabor the point, using the same analysis as given for claim 1, amended claim 18 can be seen to currently be in condition for allowance.

Claims 19-20.

Claims 19-20 ultimately depend on claim 18. Thus, at least for the reasons set forth above with respect to claim 18, claims 19-20 should also be in condition for allowance. These claims also set forth independently patentable combinations.

Claim 21.

To not belabor the point, using the same analysis as given for claim 1, amended claim 21 can be seen to currently be in condition for allowance.

Claim 27.

To not belabor the point, using the same analysis as given for claim 1, amended claim 27 can be seen to currently be in condition for allowance.

Claims 28-29.

Claims 28-29 ultimately depend on claim 27. Thus, at least for the reasons set forth above with respect to claim 27, claims 28 and 29 should also be in condition for allowance. These claims also set forth independently patentable combinations.

Claim Rejections under 35 USC § 103

The Action rejects claims 22-26 and 30 under 35 U.S.C § 103(a) as unpatentable over Pinter. Applicants respectfully disagree and traverse.

Claims 22-26.

Claims 22-26 ultimately depend on claim 21. Thus, at least for the reasons set forth above with respect to claim 21, claims 22-26 should also be in condition for allowance. This

claim also set forth independently patentable combinations over the Pinter-"design choices" combination.

Claim 30.

Claim 30 ultimately depends on claim 27. Thus, at least for the reasons set forth above with respect to claim 27, claim 30 should also be in condition for allowance. This claim also set forth independently patentable combinations over the Pinter-"design choices" combination.

Support for Amendments.

Support for the Amendments is found throughout the Specification and Figures as originally filed. In addition, we list the following support locations:

Specification, page 4, lines 5-16; page 6 line 12 to page 7 line 5; page 8, lines 5-9.

Interview Request

If the claims are not found by the Examiner to be allowable, the Examiner is requested to call the undersigned attorney to set up an interview to discuss this application.

Conclusion

The claims in their present form should be allowable. Such action is respectfully requested.

Respectfully submitted,

KLARQUIST SPARKMAN, LLP

One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Telephone: (503) 595-5300
Facsimile: (503) 595-5301

By /Genie Lyons/
Genie Lyons
Registration No. 43,841